ELSEVIER

Computational Geometry 4 (1994) 137–156

Computational Geometry
Theory and Applications

Approximate decision algorithms for point set congruence

Paul J. Heffernan[a,*,1], Stefan Schirra[b,2]

[a] *Department of Mathematical Sciences, Memphis State University, Memphis, TN 38152, USA*
[b] *Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany*

Communicated by Emo Welzl; submitted 27 August 1992; accepted 3 September 1993

Abstract

This paper considers the computer vision problem of testing whether two equal cardinality point sets A and B in the plane are ε-congruent. We say that A and B are ε-congruent if there exists an isometry I and bijection $l: A \to B$ such that $\text{dist}(I(a), l(a)) \leqslant \varepsilon$, for all $a \in A$. Since known methods for this problem are expensive, we develop approximate decision algorithms that are considerably faster than the known decision algorithms, and have bounds on their imprecision. Our approach reduces the problem to that of computing maximum flows on a series of graphs with integral capacities.

Key words: Computational geometry; Point matching; Computer vision; Approximate decision algorithms; Network flows

1. Introduction

In this paper we address the following question: given two (equal cardinality) sets of points in the plane, how do we determine whether they are approximately congruent? The practical motivation behind our study comes from computer vision, where an observed image is compared to a hypothesized model. Since the given problem appears to be difficult, we turn our attention to approximate algorithms with performance guarantees. Eventually, we reduce the problem to that of computing a maximum flow on each of a series of (s, t)-graphs. We apply known solution techniques,

*Corresponding author.

[1] Part of this work was done while the author was an NSF graduate fellow at Cornell University. Also partially supported by the U.S. Army Research Office, Grant No. DAAL03-92-G-0378.

[2] Partially supported by a grant from G.I.F., the German-Israeli Foundation for Scientific Research and Development.

Elsevier Science B.V.
SSDI 0925-7721(93)E0025-C

and a new idea of Feder and Motwani [8] for speeding-up graph algorithms, which is well suited for our max-flow problems.

In pattern recognition, one often wishes to determine how well an observed image matches a model image. For us, the model and the observed image will be planar point sets A and B of equal cardinality. In order to make A match B as closely as possible, we perform an isometry on A, and then pair each point of A with one of B such that the points of each pair lie close to each other. An *isometry* is an affine mapping in the plane that preserves distances, and is composed of a translation, rotation, and possibly a reflection. Any isometry can be represented as

$$I(x) = MJx + t,$$

where

$$M = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \text{ for some } \varphi \in [0, 2\pi), \quad J \in \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \right\},$$

and $t \in \mathbb{R}^2$. Along with an isometry I, we must designate a bijection $l: A \to B$. A natural measure of our attempt to match the observed image A with the model image B is the maximum distance between any pair $(I(a), l(a))$, $a \in A$. Because of errors in measurement, it is unrealistic to expect an observed image to match exactly the model from which it came, so we will be interested in matchings that lead to *approximate congruence*. We say that A and B are *congruent with tolerance* ε, or *ε-congruent*, if there exist an isometry I and a bijection $l: A \to B$ such that $\mathrm{dist}(I(a), l(a)) \leqslant \varepsilon$, for all $a \in A$, where $\mathrm{dist}(\cdot, \cdot)$ is the distance function for our chosen metric. We are interested in approximate congruence under the L_2 metric, with various restrictions on the isometry, such as the translation-only and rotation-only cases. The *decision problem* asks whether two point sets are ε-congruent for a given value ε, under a given metric and with a given restriction on the isometry. A *decision algorithm* for a particular metric and isometry class solves the decision problem; that is, given two point sets and a value ε, it correctly determines whether or not the two sets are ε-congruent.

Several researchers have studied approximate congruence, notably Baird [5] and Alt, et al. [2]. The distinguishing feature of their decision algorithms is the high run-time: no algorithm is known for the case of the isometry restricted to a translation with $o(n^6)$ run-time, and for the case of unrestricted isometry, the best known bound is $O(n^8)$ (this bound is tight for the algorithm of [2], as an example in this paper shows). For models with a large number of points, such performance may be unacceptable. Faced with a problem whose solution seems to be too expensive, we look for a useful solution—we trade some of the exactness for improvements in run-time, and obtain fast algorithms that come with a performance bound. We develop *approximate decision algorithms* for approximate congruence.

For a specified metric and isometry class, let $\varepsilon_{\mathrm{opt}}(A, B)$ denote the minimum value of ε such that A and B are ε-congruent. Occasionally, we will designate the isometry class implied by this notation by writing $\varepsilon^{\mathrm{T}}_{\mathrm{opt}}(A, B)$, $\varepsilon^{\mathrm{Rd}}_{\mathrm{opt}}(A, B)$, $\varepsilon^{\mathrm{J}}_{\mathrm{opt}}(A, B)$ and $\varepsilon^{\mathrm{I}}_{\mathrm{opt}}(A, B)$,

respectively, for the cases of translation, rotation about a point d, rigid motion and general isometry. Intuitively, we would believe that it is more difficult for a decision algorithm to test for ε-congruence if ε is close to $\varepsilon_{\text{opt}}(A, B)$. We would be willing to accept a decision algorithm that correctly answers queries for values of ε not near $\varepsilon_{\text{opt}}(A, B)$, but sometimes chooses not to answer when ε is near $\varepsilon_{\text{opt}}(A, B)$, if that algorithm provides substantial time savings over those decision algorithms which always return a correct answer. We call decision algorithms which always return a correct answer *complete decision algorithms*, while an algorithm which either returns a correct answer or chooses not to answer we call an *approximate decision algorithm.* An approximate decision algorithm is called (α, β)-*approximate* [21], if, for any $\varepsilon \notin [\varepsilon_{\text{opt}}(A, B) - \alpha, \varepsilon_{\text{opt}}(A, B) + \beta]$, it correctly answers a query, and for $\varepsilon \in [\varepsilon_{\text{opt}}(A, B) - \alpha, \varepsilon_{\text{opt}}(A, B) + \beta]$, it either answers correctly or chooses not to answer. We call $[\varepsilon_{\text{opt}}(A, B) - \alpha, \varepsilon_{\text{opt}}(A, B) + \beta]$ the *indecision interval.* When we say that we have an "(α, β)-approximate algorithm for testing approximate congruence", we use the word "approximate" twice, but with two different meanings. The latter instance refers to the fact that we are trying to put the points of one set within the ε-balls of the other; the former refers to the indecision interval. An (α, β)-approximate algorithm has the desirable property that it will not return an incorrect answer; if it is not sure, it will simply say that it does not know the answer.

We present algorithms for three different classes of isometries—translation, rotation with fixed center, and general isometry. While we will assume that all distances in the paper represent the Euclidean metric, our algorithms apply equally well to any L_p metric. We use an additional input parameter $\gamma \leqslant \varepsilon$ to bound the indecision interval. For ε-congruence by translation we present a (γ, γ)-approximate algorithm for testing ε-congruence that runs in time $O(n^{1.5}(\varepsilon/\gamma)^4)$. For the case of a rotation with a fixed center we present a (γ, γ)-approximate algorithm with running time $O(n^{2.5}(\varepsilon/\gamma)^3)$. Finally, we show that any (γ, γ)-approximate algorithm for rotation with a fixed center can be converted to a (γ, γ)-approximate algorithm for general isometry with an additional factor of $(\varepsilon/\gamma)^2$ appearing in the time bound, giving an algorithm for general isometry with time complexity $O(n^{2.5}(\varepsilon/\gamma)^5)$. For the general case we have a speed-up of $\Omega(n^{5.5})$ if $\gamma = \varepsilon/c$ for some constant $c \geqslant 1$. Our algorithms can be slightly modified such that functions of n in the time bound can be replaced by functions of δ/γ where $\delta = \max\{\text{diam}(A), \text{diam}(B)\}$. For example, we get a (γ, γ)-approximate algorithm for the translation case with running time $O(n + (\delta/\gamma)^4(\varepsilon/\gamma)^4 \log(\delta/\gamma))$. These alternate time bounds imply that these algorithms are especially efficient on dense data.

It is possible to remove the indecision from our approximate decision algorithms, and obtain complete decision algorithms whose time-complexity is dependent on the difficulty of the problem instance. Specifically, if we think of $K_\varepsilon = |\varepsilon_{\text{opt}}(A, B) - \varepsilon|$ as the "difficulty parameter", then each of our approximate decision algorithms can be transformed into a complete decision algorithm, with K_ε replacing γ in the time bound. For example, this gives a complete decision algorithm for testing ε-congruence under translation with time bound $O(n^{1.5}(\varepsilon/K_\varepsilon)^4)$.

As mentioned earlier, the best known algorithm for approximate congruence has running time $\Theta(n^8)$ [2]. For approximate congruence enabled by translations, several $O(n^6)$ algorithms are known [2, 11, 16]. Approximate congruence enabled by a rotation with a given center can be decided in time $O(n^4)$ [16, 21], and approximate congruence enabled by a reflection in time $O(n^6)$ [16]. An algorithm in [2] solves in time $O(n^6 \log n)$ the ε-congruence optimization problem under translation, which asks us to compute $\varepsilon^{T}_{opt}(A, B)$. $\varepsilon^{Rd}_{opt}(A, B)$ and $\varepsilon^{I}_{opt}(A, B)$ can be computed in time $O(n^4 \log n)$ and $O(n^8 \log n)$, respectively [21]. A special case of the ε-convergence problem is that of *given correspondence*, where the bijection l is given; this problem is studied in [2, 15, 16]. In [4] and [25], algorithms are given that generalize the approach of [2] by considering sets A and B of unequal cardinality, and by allowing "noise regions" which are not based on a metric. Specifically, [4] allows the "noise regions" to be arbitrary nonconvex polygons. In [3], algorithms are given for the decision problem for various classes of "noise regions" under isometry and similarity (an isometry plus a change of scale), but with the assumption that the "noise regions" around point set B are pairwise-disjoint. In [3], [4], and [25], combinatorial upper and lower bounds are given on the number of distinct bijections that can satisfy the decision problem. A problem related to ε-congruence is that of finding a translation that minimizes the Hausdorff distance between two point sets; this problem is studied in [7, 12, 13, 14]. Approximate decision algorithms have been studied by Schirra [22], who gives (γ, γ)-approximate algorithms that test for ε-congruence under translation in $O(n^{2.5}(\varepsilon/\gamma)^2)$ time and under general isometry in $O(n^4(\varepsilon/\gamma)^2)$ time, and by Heffernan [11], who gives an $O(n^3(\varepsilon/\gamma)^6)$ time translation algorithm. Behrends [6] considers approximate decision algorithms for pairwise disjoint ε-balls. We compare these results with our new results in Table 1.

Table 1
Known results and new results for ε-congruence without given correspondence of point sets of cardinality n; for the results in the second and third column of the table it is assumed that the minimum distance between points in B is at least ε. The approximate decision algorithms in the last three columns are (γ, γ)-approximate algorithms, where γ is ε divided by some constant

Decision algorithms for approximate point set congruence without given correspondence

	known results				new results
	complete decision algorithms		approximate decision algorithms		
		disjoint ε-balls			
translations	$O(n^6)$ [2, 11, 16]	$O(n \log n)$ [2]		$O(n^{2.5})$ [21, 22]	$O(n^{1.5})$
rotations	$O(n^4)$ [16, 21, 22]	$O(n^2)$ [3]			$O(n^{2.5})$
isometries	$\Theta(n^8)$ [2]	$O(n^4 \log n)$ [3]	$O(n^2 \log n)$ [6]	$O(n^4)$ [21, 22]	$O(n^{2.5})$

The remainder of the paper is organized as follows. Section 2 serves as an introduction to the tools used to construct approximate decision algorithms. In Section 3–5 we describe our algorithms for the translation, rotation, and general isometry cases. In Section 6 we give an example that motivates the development of approximate decision algorithms. For the decision algorithm for ε-congruence under general isometry of [2], we show that the worst-case time bound of $O(n^8)$ is tight. Section 7 is the conclusion.

2. Basics

Before we describe our approximate decision algorithms, we shall first discuss some of the ideas behind them. Most of the decision algorithms for approximate congruence without correspondence use the same scheme. They (1) compute a finite set of candidates for the isometry in demand, and (2) ask whether any of these candidate isometries enables approximate congruence for the planar point sets $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ with tolerance ε. Testing whether A and B are ε-congruent by an isometry I is reduced to a bipartite matching problem: two points $a \in A$ and $b \in B$ may be matched if $I(a)$ and b lie within distance ε of each other. Formally we say that isometry I enables ε-congruence for A and B if and only if the network $G(I, \varepsilon, A, B) = (\{s, t\} \cup U \cup V, E, c)$ has a max-flow of size n. Here, $U = \{u_1, \ldots, u_n\}$ represents the points in A, $V = \{v_1, \ldots, v_n\}$ represents the points in B,

$$E = \{(s, u_i) | 1 \leqslant i \leqslant n\} \cup \{(v_j, t) | 1 \leqslant j \leqslant n\} \cup \{(u_i, v_j) | \operatorname{dist}(I(a_i), b_j) \leqslant \varepsilon\},$$

and $c(e) = 1$ for all $e \in E$.

Several approaches can be used to construct the set of candidate isometries. Some algorithms (e.g. [2, 11, 21]) construct candidates that generate all possible graphs $G(I, \varepsilon, A, B)$ for the given A, B, and ε; such algorithms are complete decision algorithms, which always return an answer. Other algorithms (e.g. [21, 22]) discretize the set of isometries so that every isometry differs only slightly from some candidate. Such an approach does not necessarily generate all possible graphs $G(I, \varepsilon, A, B)$ and therefore yields an approximate decision algorithm; however, since every isometry closely resembles a candidate, the amount of imprecision is bounded.

Our algorithms use the second method described, that of discretizing the set of isometries. However, we also employ a technique, introduced in [11], of performing a "structured perturbation" of the point sets A and B in order to impose degenerate structure. We superimpose an orthogonal grid of width λ onto the plane, and call the intersection point of two lines of the grid a *grid point*. We then move each point of A and B to its nearest grid point, as illustrated in Fig. 1. We let $A^{\#} = \{a_1^{\#}, \ldots, a_n^{\#}\}$ and $B^{\#} = \{b_1^{\#}, \ldots, b_n^{\#}\}$ represent the multisets that are obtained by the structured perturbation of A and B, while $A^{\circ} = \{a_1^{\circ}, \ldots, a_h^{\circ}\}$ and $B^{\circ} = \{b_1^{\circ}, \ldots, b_m^{\circ}\}$ represent the sets of distinct points in $A^{\#}$ and $B^{\#}$. The relevant observation is that any point of the plane is within distance ε of only $O((\varepsilon/\lambda)^2)$ points of A° and B°.

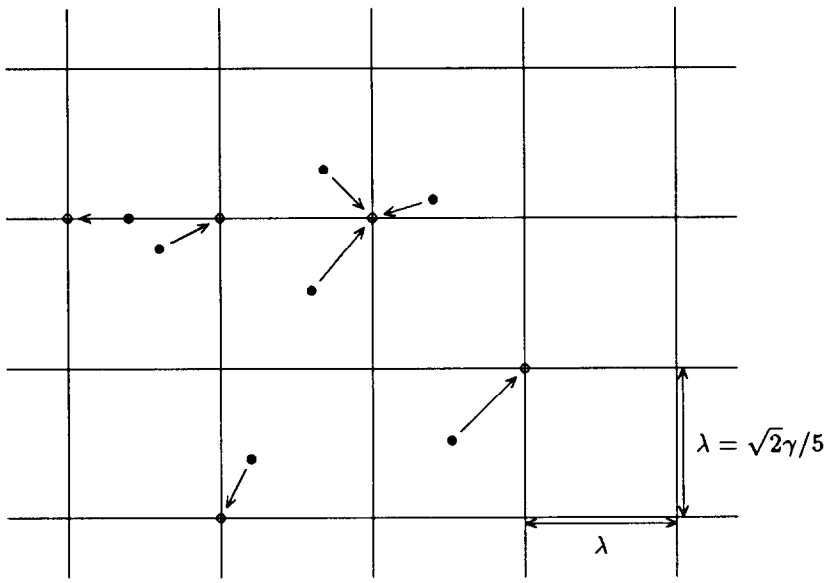

Fig. 1. Moving points to grid points.

Our algorithms proceed to test for ε-congruence of A and B by testing $A^{\#}$ and $B^{\#}$ under each of the candidates. A result of the structured perturbation, of course, is that information returned on $A^{\#}$ and $B^{\#}$ only approximates the truth about A and B; however, because the amount of perturbation is small, we can bound the imprecision. There are two sources of imprecision in our algorithms, one being the perturbation of A and B and the other discretization of the isometries. We have noted that in both cases the imprecision is bounded, which implies that the total is bounded also.

We establish efficient run-times for our algorithms by exploiting the properties of the candidate isometries and the degenerate structure of the perturbed point sets. We obtain our strongest results by a modification of the *compression graph* method of [8] to solve the max-flow problems. The compression graph method locates bipartite cliques in a graph and replaces them with simpler structures. The structure of $A^{\#}$ and $B^{\#}$ ensures that bipartite cliques in the graph $G(I, \varepsilon, A^{\#}, B^{\#})$ can be found easily, and we use this fact to efficiently replace $G(I, \varepsilon, A^{\#}, B^{\#})$ with a compression graph $G^{\#}_{\text{comp}}(I, \varepsilon, A^{\#}, B^{\#})$. We combine the threefold advantages of discretizing candidate isometries, structurally perturbing point set input, and employing compression graphs to obtain algorithms that significantly out-perform previous methods.

3. Translation

In this section we describe our algorithm for the translation case. First we state an important lemma.

Lemma 1 ([22]). *Let c_A denote the centroid of point set A and c_B denote the centroid of point set B. Let I be an isometry. If I enables ε-congruence for A and B then* $\operatorname{dist}(I(c_A), c_B) \leqslant \varepsilon$.

As stated above, our algorithms superimpose onto the plane an orthogonal grid of width λ, and perform a structured perturbation on A and B to obtain the multisets $A^{\#}$ and $B^{\#}$ and the sets of distinct points A° and B°. We choose $\lambda = \sqrt{2}\gamma/5$. We define T_{xy} as the translation that maps point x to point y. Since each point of the plane is within distance $\gamma/5$ of a grid point, we have:

Lemma 2. *If A and B are μ-congruent by a translation then there is a grid point g with* $\operatorname{dist}(g, c_B) \leqslant \mu + \gamma/5$ *such that $T_{c_A g}$ enables approximate congruence with tolerance $\mu + \gamma/5$ for A and B.*

As each point is perturbed by a distance of at most $\gamma/5$, we have the following.

Lemma 3. *If A and B are μ-congruent by a translation then $A^{\#}$ and $B^{\#}$ are approximately congruent with tolerance $\mu + 2\gamma/5$ by that translation.*

Combining Lemma 2 and Lemma 3 we get the following lemma.

Lemma 4. *If A and B are μ-congruent by a translation then there is a grid point g with* $\operatorname{dist}(g, c_B) \leqslant \mu + \gamma/5$ *such that $T_{c_A g}$ enables approximate congruence with tolerance $\mu + 3\gamma/5$ for $A^{\#}$ and $B^{\#}$.*

Hence we can conclude that A and B are not ε-congruent if for all grid points g with $\operatorname{dist}(g, c_B) \leqslant \varepsilon + \gamma/5$, the translation $T_{c_A g}$ does not enable approximate congruence with tolerance $\varepsilon + 3\gamma/5$ for $A^{\#}$ and $B^{\#}$. On the other hand we have the following.

Lemma 5. *If $A^{\#}$ and $B^{\#}$ are approximately congruent with tolerance μ by $T_{c_A g}$ for some grid point g then A and B are approximately congruent with tolerance $\mu + 2\gamma/5$ by that translation.*

Lemmas 4 and 5 form the basis of a (γ, γ)-approximate algorithm, which is given in Table 2. By Lemma 5, the point sets A and B are approximately congruent with tolerance ε by a translation if $A^{\#}$ and $B^{\#}$ are approximately congruent with tolerance $\varepsilon - 2\gamma/5$ by $T_{c_A g}$ for some g; thus, any YES answer is correct. Lemma 4 states that if there is no grid point g among those considered such that $T_{c_A g}$ enables congruence with tolerance $\varepsilon + 3\gamma/5$, then A and B are not ε-congruent; thus, a NO answer must be correct. We see that if the algorithm in Table 2 gives an answer, then that answer is correct.

Table 2
(γ, γ)-Approximate algorithm for ε-congruence by translations

```
1   Compute A# and B# from A and B;
2   𝒞 = {T_{c_A g} | dist(g, c_B) ≤ ε + γ/5 and g is grid point};
3   possible = false;
4   for all I ∈ 𝒞
5   do if I leads to (ε − 2γ/5)-congruence of A# and B#
6     then return YES fi;
7     if I leads to (ε + 3γ/5)-congruence of A# and B#
8     then possible = true fi od;
9   if possible
10  then return DO NOT KNOW
11  else return NO fi.
```

Lemma 6. *The algorithm in Table 2 is a (γ, γ)-approximate algorithm for approximate congruence with tolerance ε enabled by a translation.*

Proof. We argued above that if the algorithm returns an answer, then that answer is correct. We must show that the algorithm returns an answer for any ε outside of the indecision interval. Let $\varepsilon < \varepsilon^{\mathrm{T}}_{\mathrm{opt}}(A, B) - \gamma$. Then A and B are not approximately congruent with tolerance $\varepsilon + \gamma$. Hence there is no grid point g such that $T_{c_A g}$ enables approximate congruence with tolerance $\varepsilon + 3\gamma/5$ for $A^{\#}$ and $B^{\#}$, because this would imply that A and B are approximately congruent with tolerance $\varepsilon + 3\gamma/5 + 2\gamma/5$ by Lemma 5. Now let $\varepsilon \geqslant \varepsilon^{\mathrm{T}}_{\mathrm{opt}}(A, B) + \gamma$. Then A and B are approximately congruent with tolerance $\varepsilon - \gamma$. By Lemma 4 there is a grid point g such that $A^{\#}$ and $B^{\#}$ are approximately congruent with tolerance $\varepsilon - \gamma + 3\gamma/5 = \varepsilon - 2\gamma/5$ enabled by $T_{c_A g}$. □

It remains to discuss how to test whether $T_{c_A g}$ enables μ-congruence for $A^{\#}$ and $B^{\#}$. As we said earlier, we will use a max-flow algorithm to test for congruence. To represent graph networks, we will use the notation (V, E, c), where V and E are the vertex and edge sets, respectively, and $c: E \to \mathbb{N}$ gives the edge capacities. Consider the network

$$G(T_{c_A g}, \mu, A^{\#}, B^{\#}) = (\{s, t\} \cup U \cup V, E, c)$$

where $U = \{u_1, \ldots, u_n\}$ represents the points in $A^{\#}$ and $V = \{v_1, \ldots, v_n\}$ represents the points in $B^{\#}$,

$$E = \{(s, u_i) \mid 1 \leqslant i \leqslant n\} \cup \{(v_j, t) \mid 1 \leqslant j \leqslant n\} \cup \{(u_i, v_j) \mid \mathrm{dist}(T_{c_A g}\,(a_i^{\#}), b_j^{\#}) \leqslant \mu\},$$

and

$$c(e) = 1 \quad \text{for every } e \in E.$$

If $G(T_{c_A g}, \mu, A^{\#}, B^{\#})$ has a max-flow of size n, then $T_{c_A g}$ enables μ-convergence for $A^{\#}$ and $B^{\#}$. However, in the worst case the number of edges in the graph can be $\Theta(n^2)$.

Using standard techniques [19, 26] this gives a $O(n^{2.5})$ worst-case time-bound for each test.

Recently Feder and Motwani [8] presented a method for speeding-up graph algorithms. They search for complete bipartite subgraphs and substitute simpler structures for them. The resulting graphs are called *compressed*. With their technique a max-flow for graph $G(T_{c_A g}, \mu, A^{\#}, B^{\#})$ can be computed in time $O(n^{2.5}/\log n)$.

The special structure of $A^{\#}$ and $B^{\#}$ actually allows us to enjoy even greater savings with the compression graph. Let $U_k^{\circ} = \{u_i | a_i \text{ is moved onto } a_k^{\circ}\} \subset U, k = 1, \ldots, h$, and $V_l^{\circ} = \{v_j | b_j \text{ is moved onto } b_l^{\circ}\} \subset V, l = 1, \ldots, m$. If $T_{c_A g}$ moves grid point a_k° into the μ-neighborhood of b_l°, this generates a complete bipartite subgraph with node sets U_k° and V_l°. For every such bipartite clique, we remove the edges in $U_k^{\circ} \times V_l^{\circ}$, add a new node w, and add edges $\{(u_i, w) | u_i \in U_k^{\circ}\}$ and $\{(w, v_j) | v_j \in V_l^{\circ}\}$ with capacity 1 each. Thereby we replace $|U_k^{\circ}| \cdot |V_l^{\circ}|$ old edges by $|U_k^{\circ}| + |V_l^{\circ}|$ new edges. We obtain the network

$$G^{\#}_{\text{comp}}(T_{c_A g}, \mu, A^{\#}, B^{\#}) = (\{s, t\} \cup U \cup V \cup W, E, c)$$

where $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$ represent the points of $A^{\#}$ and $B^{\#}$, respectively,

$$W = \{w^{\otimes}_{k,l} | \operatorname{dist}(T_{c_A g}(a_k^{\circ}), b_l^{\circ}) \leqslant \mu\}$$

represents complete bipartite subgraphs in $G(T_{c_A g}, \mu, A^{\#}, B^{\#})$,

$$E = \{(s, u_i) | 1 \leqslant i \leqslant n\} \cup \{(v_j, t) | 1 \leqslant j \leqslant n\} \cup \{(u_i, w^{\otimes}_{k,l}) | a_i^{\#} = a_k^{\circ}, w^{\otimes}_{k,l} \in W\}$$
$$\cup \{(w^{\otimes}_{k,l}, v_j) | b_j^{\#} = b_l^{\circ}, w^{\otimes}_{k,l} \in W\},$$

and $c(e) = 1$ for all $e \in E$. A max-flow in $G^{\#}_{\text{comp}}(T_{c_A g}, \mu, A^{\#}, B^{\#})$ corresponds to a max-flow in $G(T_{c_A g}, \mu, A^{\#}, B^{\#})$. With Dinic's algorithm a max-flow in $G^{\#}_{\text{comp}}(T_{c_A g}, \mu, A^{\#}, B^{\#})$ can be computed in $O(\sqrt{n})$ phases, each in time proportional to the new number of edges. (The reason is that Dinic's algorithm takes $O(\sqrt{\text{number of nodes}})$ phases in a simple 0–1 network. A node is called simple, if it has indegree 1 or outdegree 1, and a network is called simple, if all nodes are simple. Note that the nodes in W are not simple in the compression of a graph. Analogously to the proof on the number of phases of Dinic's algorithm for simple 0–1 networks in [19, p. 81] it can be shown that Dinic's algorithm takes $O(\sqrt{\text{number of simple nodes}})$ phases in a 0–1 network, if no edge connects two non-simple nodes.)

The μ-neighborhood of any point contains $O((\mu/\gamma)^2)$ grid points. Hence each point of $A^{\#} \cup B^{\#}$ contributes to $O((\mu/\gamma)^2)$ bipartite cliques. Hence the number of edges in network $G^{\#}_{\text{comp}}(T_{c_A g}, \mu, A^{\#}, B^{\#})$ is $O(n(\mu/\gamma)^2)$. Thus Dinic's algorithm takes $O(\sqrt{n} \cdot n(\mu/\gamma)^2)$ time.

For the construction of $G^{\#}_{\text{comp}}(T_{c_A g}, \mu, A^{\#}, B^{\#})$ we use range searching. For a set of n points a data structure of size $O(n)$ can be built in $O(n \log n)$ time for fixed radius disk queries, such that each query has time complexity $O(\log n + k)$, where k is the size

of the output [17]. For each point a° in A° we ask for the points of B° contained in the disks with center $T_{c_A g}(a^\circ)$ and with radii $\mu = \varepsilon - 2\gamma/5$ and $\mu = \varepsilon + 3\gamma/5$. We have output size $k = O((\mu/\gamma)^2)$ for each query. Summing over all queries gives run-time $O(n(\log n + (\mu/\gamma)^2))$.

If space is not important, we could alternatively use standard range searched [20] with squares of side length 2μ containing the query disks and test each reported point for inclusion in the query disk. This approach has the same time bounds for the two dimensional queries considered here, but space complexity $O(n \log n)$. However, it can be used in higher dimensions, too. For queries in $\mathbb{R}^d$, preprocessing time is $O(n(\log n)^{d-1})$, query time $O((\log n)^{d-1} + k)$, and space complexity $O((n(\log n)^{d-1})$ [20].

Theorem 1. *The algorithm in Table 2 has running time* $O(n^{1.5}(\varepsilon/\gamma)^4)$.

Proof. $A^\#$ and $B^\#$ can be constructed in $O(n)$ time, and there are $O((\varepsilon/\gamma)^2)$ grid points g for which $T_{c_A g}$ is tested. Each network $G^\#_{\text{comp}}(T_{c_A g}, \mu, A^\#, B^\#)$ can be constructed in time $O(n \log n + n(\varepsilon/\gamma)^2)$ and a max-flow can be computed in time $O(\sqrt{n} \cdot n(\varepsilon/\gamma)^2)$. □

We now give an alternative for testing whether $T_{c_A g}$ enables μ-congruence. We denote the number of points in A moved onto a_i° by n_i^A and the number of points in B moved to b_j° by n_j^B. We consider the following network

$$G^\circ(T_{c_A g}, \mu, A^\circ, B^\circ) = (\{s, t\} \cup \{u_1, \ldots, u_h\} \cup \{v_1, \ldots, v_m\},\ E^\circ, c)$$

where $\{u_1, \ldots, u_h\}$ represents the points in A° and $\{v_1, \ldots, v_m\}$ represents the points in B°,

$$E^\circ = \{(s, u_i) \mid 1 \leqslant i \leqslant h\} \cup \{(v_j, t) \mid 1 \leqslant j \leqslant m\} \cup \{(u_i, v_j) \mid \text{dist}(T_{c_A g}(a_i^\circ), b_j^\circ) \leqslant \mu\},$$

and

$$c(s, u_i) = n_i^A, \quad c(v_j, t) = n_j^B, \quad c(u_i, v_j) = \infty.$$

We claim that it suffices to compute a max-flow on $G^\circ(T_{c_A g}, \mu, A^\circ, B^\circ)$.

Lemma 7. *$G^\circ(T_{c_A g}, \mu, A^\circ, B^\circ)$ has a flow of capacity n iff the multisets $A^\#$ and $B^\#$ are μ-congruent by $T_{c_A g}$.*

Proof. It is easy to see that a labeling implies a flow of capacity n. For the converse, consider the well-known fact that there exists an integral max-flow solution if all edge capacities are integral [19, 26]. Hence if there is a max-flow with capacity n, then there is a max-flow of capacity n with integer flow on each edge. Now it is easy to derive a labeling $l: A^\# \to B^\#$ from such a flow. Consider the edges one by one. If the max-flow pushes f units over edge (u_i, v_j) then l maps f of the unused copies of a_i° to f unused copies of b_j°. Then these copies are marked as used and the next edge is

considered. The capacities in the network guarantee that the number of unused copies is always sufficient. □

It remains to establish the time-complexity of computing a max-flow on a graph G° generated by the algorithm. There is a wealth of literature on the max-flow problem (for an overview, see [1, 9, 10, 26]), with numerous algorithms whose complexities depend principally on N and M, the number of nodes and edges, respectively, of G°. We will establish bounds on N and M, and state the complexities that we can obtain with max-flow algorithms. Again we use the fact that the ε-ball around a point of A° or B° contains only $O((\varepsilon/\gamma)^2)$ grid points, so a node of G° can be incident to only this many edges. If we use the trivial bound of $2n$ for N, then $M = O(n(\varepsilon/\gamma)^2)$. An alternate bound on N arises by considering the diameters of A and B. If $\delta = \max\{\text{diam}(A), \text{diam}(B)\}$, then there can be only $O((\delta/\gamma)^2)$ points of A° and B°, giving that N has this bound, and $M = O((\delta/\gamma)^2(\varepsilon/\gamma)^2)$.

The method of Sleator and Tarjan [23, 24] computes max-flows in time $O(NM \log N)$, which yields $O((\delta/\gamma)^4(\varepsilon/\gamma)^2 \log(\delta/\gamma))$. The graph G° can be built from A° and B° in $O((\delta/\gamma)^2(\varepsilon/\gamma)^2)$ time, since this is the number of potential edges that must be checked; therefore the time to build a graph is dominated by the time to find its max-flow. Since A° and B° can be constructed from A and B in time $O(n + (\delta/\gamma)^2)$ (by a type of bucket-sort), we obtain a bound of $O(n + (\delta/\gamma)^5(\varepsilon/\gamma)^4 \log(\delta/\gamma))$, which is better than the one given earlier when $n = \Omega((\delta/\gamma)^{8/3}(\log(\delta/\gamma))^{2/3})$. We summarize these in Table 3.

The max-flow methods mentioned here compute an integral maximum flow, if the input graph has integral capacities. If a max-flow of size n is computed, it is, therefore, an integral flow, and corresponds to a matching of the point sets A and B. This implies that we solve something more than the decision problem: if we determine that the point sets are ε-congruent, we actually return a bijection l and a translation T that enable ε-congruence.

We mention here that our results for the translation case extend easily to higher dimensions. We showed in this section that a (γ, γ)-approximate algorithm can be obtained by considering a set of points representing candidate translations, such that no point within distance ε of c_B is more than distance $\gamma/5$ from a candidate point. If we cover $\mathbb{R}^d$ with orthogonal grid points at width λ, then no point of $\mathbb{R}^d$ is more than distance $\sqrt{d}\lambda/2$ from a grid point. By setting $\sqrt{d}\lambda/2 = \gamma/5$, we see that we should choose $\lambda = 2\gamma/(5\sqrt{d})$ (note that for $d = 2$, this gives the value $\lambda = 2\gamma/(5\sqrt{2})$ used in

Table 3
Time-bounds for the (γ, γ)-approximate algorithms under translation

graph	total time	range
$G^{\neq}_{\text{comp}}$	$O(n^{1.5}(\varepsilon/\gamma)^4)$	$n = o((\delta/\gamma)^{8/3}(\log(\delta/\gamma))^{2/3})$
G°	$O(n + (\delta/\gamma)^4(\varepsilon/\gamma)^4 \log(\delta/\gamma))$	$n = \Omega((\delta/\gamma)^{8/3}(\log(\delta/\gamma))^{2/3})$

this section). By Lemma 2, we need only consider translations that map c_A to a grid point within distance $\varepsilon + \gamma/5$ of c_B. There are $O((\varepsilon/\lambda)^d)$ such points, which for $d = O(1)$ gives:

Theorem 2. *The algorithm in Table 2 is a (γ, γ)-approximate algorithm for ε-congruence under translation for point sets in $\mathbb{R}^d$. The algorithm runs in time $O(n^{1.5}(\varepsilon/\gamma)^{2d})$ resp. $O(n + (\delta/\gamma)^{2d}(\varepsilon/\gamma)^{2d}\log(\delta/\gamma))$.*

4. Fixed center rotations

Approximate decision algorithms for rotation differ from those for translation in only one respect: the method of computing candidate isometries. This difference is a consequence of the contrasting natures of the two types of isometries. Again we superimpose an orthogonal grid with separation $\lambda = \sqrt{2}\gamma/5$ onto the plane and move each point in A and B to its nearest grid point to form the multisets $A^{\#}$ and $B^{\#}$, and the sets of distinct points A° and B°. We let a°_{far} be the point in A° farthest from the fixed rotation center d and $\bigcirc_{far}$ be the circle of radius $\mathrm{dist}(d, a^{\circ}_{far})$ and center d. (cf. Fig. 2). We partition $\bigcirc_{far}$ into arcs of length roughly $2\gamma/5$. We let P be the set of endpoints of these arcs and $P^{\circ} = \{p \in P;\ \exists b^{\circ} \in B^{\circ} \text{ with } \mathrm{dist}(p, b^{\circ}) \leqslant \varepsilon + 3\gamma/5\}$. For each point $o \in \bigcirc_{far}$ $\mathrm{dist}(o, b^{\circ}) \leqslant \varepsilon + 2\gamma/5$ for some $b^{\circ} \in B^{\circ}$ there is a point $p \in P^{\circ}$ with $\mathrm{dist}(o, p) \leqslant \gamma/5$.

The set of candidate rotations consists of those rotations around d which map a°_{far} onto a point in P°. All other rotations can be ignored, because they fail to map a°_{far} close (i.e. within distance $\varepsilon + 3\gamma/5$) of any point of B°. For a point $p \in \bigcirc_{far}$ let

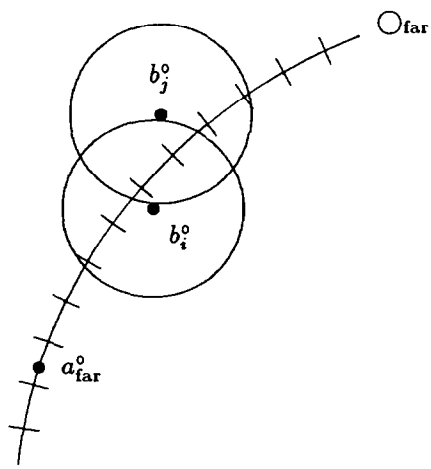

Fig. 2. $\bigcirc_{far}$.

$R_{a^\circ_{far}p}$ be the rotation with center d, which maps a°_{far} onto p. Because each point of A and B is perturbed at most $\gamma/5$ in the formation of $A^\#$ and $B^\#$, we have the following.

Lemma 8. *If $A^\#$ and $B^\#$ are approximately congruent with tolerance ε by $R_{a^\circ_{far}p}$ for some point $p \in P^\circ$, then A and B are approximately congruent with tolerance $\varepsilon + 2\gamma/5$ by that rotation.*

If A and B are ε-congruent under a certain rotation R, then $A^\#$ and $B^\#$ are $\varepsilon + 2\gamma/5$-congruent under R. There is a point $b^\circ \in B^\circ$ within distance $\varepsilon + 2\gamma/5$ of $R(a^\circ_{far})$, and $R(a^\circ_{far})$ is within distance $\gamma/5$ of a point $p \in P$, yielding $\mathrm{dist}(b^\circ, p) \leqslant \varepsilon + 3\gamma/5$. This point p is in P° by the definition of P°. Because all points of $A^\#$ lie inside $\bigcirc_{far}$, the distance between $R(a^\#)$ and $R_{a^\circ_{far}p}(a^\#)$ is less than $\gamma/5$ for all $a^\# \in A^\#$. Therefore $R_{a^\circ_{far}p}$ enables $\varepsilon + 3\gamma/5$-congruence for $A^\#$ and $B^\#$. This is summarized below.

Lemma 9. *If A and B are ε-congruent by a rotation with center d, then there is a point $b^\circ \in B^\circ$ and a point $p \in P^\circ$ with $\mathrm{dist}(p, b^\circ) \leqslant \varepsilon + 3\gamma/5$ such that $R_{a^\circ_{far}p}$ enables approximate congruence with tolerance $\varepsilon + 3\gamma/5$ for $A^\#$ and $B^\#$.*

Lemmas 8 and 9 are the rotation analogues of Lemmas 5 and 4 from the translation section. It follows that an identical algorithm for ε-congruence exists for rotation, the only difference being the set of candidates. The resulting (γ, γ)-approximate algorithm of Table 4 differs from the one given for translations in Table 2 only in line 2.

To test a candidate rotation $R_{a^\circ_{far}p}$ for μ-congruence, we compute a max-flow for the network $G^\#_{comp}(R_{a^\circ_{far}p}, \mu, A^\#, B^\#)$. Of course, the options for computing a max-flow on $G^\#_{comp}(R_{a^\circ_{far}p}, \mu, A^\#, B^\#)$ are the same as we encountered in the translation case. The time-complexities of the rotation algorithms vary only on account of the size of the candidate set. A given point $b^\circ \in B^\circ$ is within distance $\varepsilon + 3\gamma/5$ of only $O(\varepsilon/\gamma)$ points $p \in P$, which means that $|\mathscr{C}| = O(n\varepsilon/\gamma)$. Employing [8] we get the following.

Table 4
(γ, γ)-Approximate algorithm for ε-congruence by rotations

```
1   Compute A# and B# from A and B;
2   C = {R_{a°far p} | p ∈ P°};
3   possible = false;
4   for all I ∈ C
5   do if I leads to (ε − 2γ/5)-congruence of A# and B#
6     then return YES fi;
7     if I leads to (ε + 3γ/5)-congruence of A# and B#
8     then possible = true fi od;
9   if possible
10  then return DO NOT KNOW
11  else return NO fi.
```

Table 5
Time-bounds for the (γ, γ)-approximate algorithms under fixed center rotations

graph	total time	range
$G^{\neq}_{\text{comp}}$	$O(n^{1.5} \min(n\varepsilon, \delta) \cdot \varepsilon^2/\gamma^3)$	$n = o((\delta/\gamma)^{8/3}(\log(\delta/\gamma))^{2/3})$
G°	$O(n + (\delta/\gamma)^5(\varepsilon/\gamma)^2 \log(\delta/\gamma))$	$n = \Omega((\delta/\gamma)^{8/3}(\log(\delta/\gamma))^{2/3})$

Theorem 3. *ε-congruence of two point sets of cardinality n under rotations with a fixed center can be decided by a (γ, γ)-approximate algorithm in time $O(n^{2.5}(\varepsilon/\gamma)^3)$.*

Alternate bounds are attainable if we note that $|\mathscr{C}|$ can be expressed as $O(\delta/\gamma)$, since the circumference of $\bigcirc_{\text{far}}$ is $O(\delta)$ and the candidate points are spaced $O(\gamma)$ apart. The bound on the method based on [8] can be rephrased as $O(n^{1.5} \min(n\varepsilon, \delta) \cdot \varepsilon^2/\gamma^3)$. The Sleator-Tarjan method [23, 24] yields $O(n + (\delta/\gamma)^5 (\varepsilon/\gamma)^2 \log(\delta/\gamma))$. The results are summarized in Table 5. The comparative ranges of the algorithms have been derived, keeping in mind that $\delta/\varepsilon < (\delta/\gamma)^{8/3}(\log(\delta/\gamma))^{2/3}$, because $\gamma < \varepsilon$.

Another time-bound can be obtained by using $|\mathscr{C}| = O(n\varepsilon/\gamma)$ with the Sleator–Tarjan method, but it can be shown that it is dominated by the above bounds.

5. General case

As is noted in [2], a decision algorithm for approximate congruence enabled by rigid motions, i.e., isometries without reflection, is sufficient to decide approximate congruence enabled by an arbitrary isometry: one merely tests A and B as well as A and B' for approximate congruence under a rigid motion, where B' is the image of B under some arbitrarily chosen reflection. For approximate congruence enabled by rigid motions, the best-known complete decision algorithm [2] has running time $O(n^8)$. The time bound is asymptotically tight, i.e., the running time is $\Omega(n^8)$ in the worst case, as we shall see in the next section. So approximate decision algorithms are particularly interesting in this case. In [22] a (γ, γ)-approximate algorithm with running time $O(n^4(\varepsilon/\gamma)^2)$ has been presented. In this section we show that any (γ, γ)-approximate algorithm for rotation with a fixed center can be converted to a (γ, γ)-approximate algorithm for rigid motion (and therefore to one for general isometry) with an additional factor of $(\varepsilon/\gamma)^2$ appearing in the time bound. As a result, we are able to improve upon the bound of [22], with an algorithm for general isometry with time complexity $O(n^{2.5}(\varepsilon/\gamma)^5)$. For dense data, we have algorithms with bounds $O(n^{1.5}(\delta/\gamma)(\varepsilon/\gamma)^4)$ and $O(n + (\delta/\gamma)^5(\varepsilon/\gamma)^4 \log(\delta/\gamma))$. The algorithms are based on

Lemma 10. *Let J be a rigid motion and l a labeling that enable approximate congruence with tolerance μ for A and B. Let $T_{c_A J(c_A)}$ be the translation that maps c_A onto $J(c_A)$. Then*

there is a rotation R with center $J(c_A)$ such that R and l enable approximate congruence with tolerance μ for $T_{c_A J(c_A)}(A)$ and B.

Proof. Every rigid motion can be composed of a rotation around an arbitrary center and a suitable translation [18]. □

The previous lemma immediately gives

Lemma 11. *Let J be a rigid motion and l a labeling that enable approximate congruence with tolerance μ for A and B. Let d be an arbitrary point. Let $T_{c_A d}$ be the translation that maps c_A onto d. There is a rotation centered at d such that R and l enable approximate congruence with tolerance $\mu + \operatorname{dist}(d, J(c_A))$ for $T_{c_A d}(A)$ and B.*

Let J_{opt} be a rigid motion and l a labelling which enable ε-congruence for A and B. By Lemma 1, J_{opt} maps c_A into the ε-neighborhood of c_B. The algorithm in Table 6 inspects rotation centers in the ε-neighborhood of c_B and uses an (α, β)-approximate algorithm for testing approximate congruence under fixed center rotations. The set of rotation centers is chosen such that for each point in the ε-neighborhood of c_B there is rotation center within distance v.

Lemma 12. *The algorithm in Table 6 is an $(\alpha + v, \beta + v)$-approximate algorithm for approximate congruence under rigid motions.*

Proof. First we prove the correctness of the algorithm. If $\mathscr{R}(T_{c_A g}(A), B, g, \varepsilon)$ returns YES for some g then A and B are ε-congruent, too. If $\mathscr{R}(T_{c_A g}(A), B, g, \varepsilon + v)$ returns NO for all $g \in L$ then A and B cannot be ε-congruent by Lemma 11.

Next we show that the algorithm is $(\alpha + v, \beta + v)$-approximate. Let $\varepsilon < \varepsilon^{\text{J}}_{\text{opt}}(A, B) - (\alpha + v)$. For all $g \in L$ we have $\varepsilon^{\text{Rg}}_{\text{opt}}(T_{c_A g}(A), B) \geq \varepsilon^{\text{J}}_{\text{opt}}(A, B)$. Hence $\varepsilon + v < \varepsilon^{\text{Rg}}_{\text{opt}}(T_{c_A g}(A), B) - \alpha$ for all g. So $\mathscr{R}(T_{c_A g}(A), B, g, \varepsilon + v)$ returns NO for all

Table 6
$(\alpha + v, \beta + v)$-Approximate algorithm for ε-congruence by rigid motions

```
1   Let R(A, B, c, ε) be an (α, β)-approximate algorithm for approximate congruence with tolerance ε for
    point sets A and B under rotations with fixed center c;
2   Choose a set L of points such that each point in the ε-neighborhood of c_B has distanc at most v to its
    nearest point in L;
3   possible = false;
4   for all g ∈ L
5   do if R(T_{c_A g}(A), B, g, ε) returns YES
6     then return YES fi;
7     if R(T_{c_A g}(A), B, g, ε + v) returns YES or DO NOT KNOW
8     then possible = true fi od;
9   if possible
10  then return DO NOT KNOW
11  else return NO fi.
```

g because it is $(\alpha, \cdot)$-approximate. Now let $\varepsilon \geq \varepsilon^{J}_{opt}(A, B) + \beta + \nu$. There is a $g \in L$ such that $\varepsilon^{Rg}_{opt}(T_{c_A g}(A), B) \leq \varepsilon^{J}_{opt}(A, B) + \nu$ by Lemma 1 and Lemma 11. Therefore $\varepsilon \geq \varepsilon^{Rg}_{opt}(T_{c_A g}(A), B) + \beta$ and $\mathcal{R}(T_{c_A g}(A), B, g, \varepsilon)$ returns YES for this g because it is $(\cdot, \beta)$-approximate. Hence the algorithm in Table 6 is $(\alpha + \nu, \beta + \nu)$-approximate because $\mathcal{R}$ is (α, β)-approximate. □

The run-time of the algorithm is $O(|L| \cdot$ run-time of algorithm $\mathcal{R})$. L can be chosen such that $|L| = O((\varepsilon/\nu)^2)$. By choosing $\alpha = \beta = \nu = \gamma/2$ and using the rotation algorithms of the previous section, we get (γ, γ)-approximate algorithms for approximate congruence under rigid motions and therefore for approximate congruence under general isometries, also.

Theorem 4. *There is a (γ, γ)-approximate algorithm for approximate congruence enabled by a general isometry with running time $O(n^{2.5}(\varepsilon/\gamma)^5)$. Let $\delta = \max(\mathrm{diam}(A), \mathrm{diam}(B))$. There are (γ, γ)-approximate algorithms for approximate congruence enabled by a general isometry with running time $O(n^{1.5} \min(n\varepsilon, \delta) \cdot \varepsilon^4/\gamma^5)$ and $O(n + (\delta/\gamma)^5 (\varepsilon/\gamma)^4 \log(\delta/\gamma))$.*

In the last time-bound of the above theorem, the term n has not been multiplied by $(\varepsilon/\gamma)^2$, since n appears only as a result of the preprocessing step of constructing A° and B° from A and B.

6. Lower bound on the worst case running time of the decision algorithm of Alt et al.

We want to show that the algorithm of Alt et al. [2] for deciding ε-congruence of two sets $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$ of n points in the plane enabled by a rigid motion has worst case running time $\Theta(n^8)$. We start with a brief description of the algorithm.

If there is a rigid motion that enables ε-congruence, then there is a rigid motion that enables ε-congruence and maps a_i and a_j onto circles C_k and C_l with radius ε and center b_k and b_l resp. for some $i, j, k, l \in \{1, \ldots, n\}$. In the algorithm of [2] such an isometry is searched for among all combinations of i, j, k, and l. Let i, j, k, l be fixed. If a_i and a_j are simultaneously moved on C_k and C_l resp., the other points of A move on algebraic curves [2, 21] which possibly intersect the circles with radius ε centered at the points in B. If the motion of a_i and a_j on C_k and C_l is parametrized we get a set $I_{m,p}$ of parameter values for each pair $(m, p) \in \{1, \ldots, n\}^2$ such that a_m has distance at most ε to b_p for the parameter values in $I_{m,p}$. Each $I_{m,p}$ consists of $O(1)$ intervals. The endpoints of these intervals are sorted and the isometric mappings corresponding to these parameter values are tested for enabling ε-congruence. For the test whether rigid motion J enables approximate congruence with tolerance ε, a maximum matching in $G_J = (U \cup V, E_J)$ is computed, where $U = \{u_1, \ldots, u_n\}$ represents the points in A, $V = \{v_1, \ldots, v_n\}$ represents the points in B, and $E_J = \{\{u_s, v_t\} \mid \mathrm{dist}(J(a_s), b_t) \leq \varepsilon\}$.

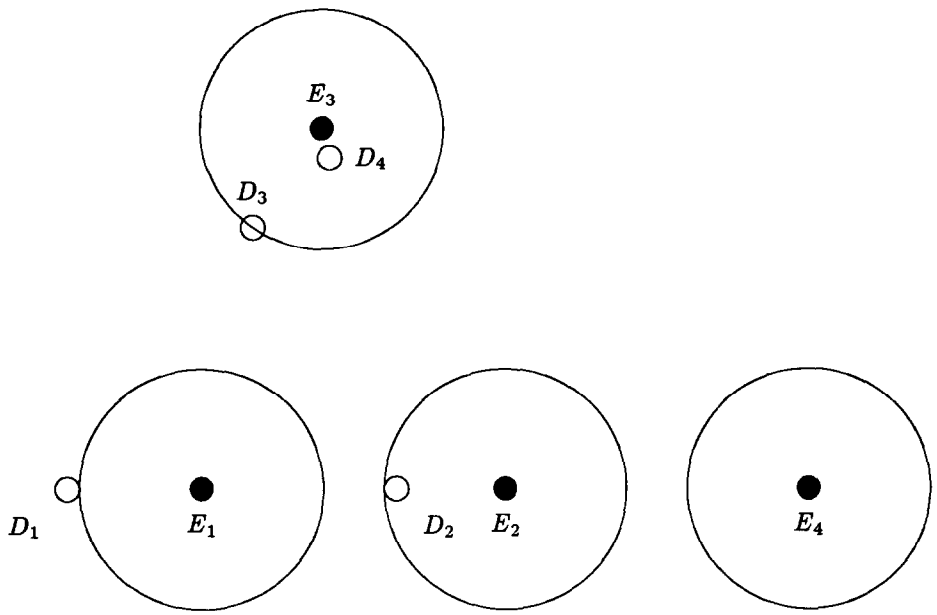

Fig. 3. $n/4$ points are in D_1, D_2, D_3, D_4, E_1, E_2; $n/8$ points are in E_3; and $3n/8$ points are in E_4. The large circles are ε-balls centered at the midpoints of E_i.

At first the graph and a maximum matching are computed for the isometry corresponding to the smallest parameter value. The graph and maximum matching for the other isometries are computed one by one, according to increasing parameter values, using the graph and maximum matching of the preceeding isometry. The maximum matching is updated by a depth first search for an augmenting path in the residual graph started in the unmatched nodes. The algorithm stops if a perfect matching has been found.

For a lower bound on the worst case running time of this algorithm consider Fig. 3. Let $n/4$ of the points of B be in the circles $D_1, \ldots, D_4$ each. $n/4$ of the points of A are in E_1 and E_2 each. Furthermore $n/8$ points are in E_3 and $3n/8$ points are in E_4. We assume that the points are in general position, i.e. all distances between different point pairs are different. Since A and B are not ε-congruent, all combinations of i, j, k, l are taken into consideration. We concentrate on those $\Omega(n^4)$ combinations where a_i is in D_1, a_j in D_2, b_k in E_1, and b_l in E_2. When a_i and a_j are moved on C_k and C_l resp., each point of A which is in D_3 moves into the ε-neighborhood of each point of B which is in E_3. Consider the $\Omega(n^2)$ interval endpoints corresponding to these events. At each such event a new edge, say $\{u_x, v_y\}$, is added to the present graph, where u_x corresponds to a point in D_3 and v_y corresponds to a point in E_3. The $n/8$ nodes corresponding to the points in E_3 and the $n/4$ nodes corresponding to the points in D_4 form a complete bipartite subgraph. Since we maintain a maximum matching, all nodes corresponding to the points in E_3 are already matched, when the new edge is added. Furthermore, the set of nodes reachable from u_x and v_y is contained in the set of nodes corresponding to

the points in $D_3 \cup D_4 \cup E_3$. Hence the new edge cannot give rise to an augmenting path. Since $\Omega(n)$ of the nodes corresponding to the points in D_4 are unmatched, and each of these nodes has $\Omega(n)$ incident edges, depth first search takes time $\Omega(n^2)$. Since there are $\Omega(n^4)$ such events for each of the $\Omega(n^4)$ combinations, the overall running time is $\Omega(n^8)$. Alt et al. have shown that the running time is $O(n^8)$, so the worst case running time is $\Theta(n^8)$.

The example above is a good example for the utility of our approximate decision algorithms in Table 4 and Table 6. Started with $\gamma = \varepsilon/4$ and $\nu = \varepsilon/4$, we get a NO answer in time $O(n^{1.5})$: The points in A and B are moved to at most 32 different points. $A^{\#}$, $B^{\#}$, A°, B° can be computed in time $O(n \log n)$. There are $O(1)$ grid points in the neighborhood of the centroid of B. For each such grid point, the number of graphs considered in the fixed center rotation algorithm is $O(1)$. Each of these graphs is of size $O(n)$ and can be computed in time $O(n)$. Thus a max-flow can be computed in time $O(n\sqrt{n})$. So in this extreme example we have a speedup of $n^{6.5}$ compared to the complete decision algorithm of Alt et al.

7. Conclusion

In this paper we have addressed the question of testing whether two equal-cardinality point sets are ε-congruent, under the general L_p metric and a specified class of isometries. Our algorithms are (γ, γ)-approximate, which means that, for any $\varepsilon \notin [\varepsilon_{\text{opt}}(A, B) - \gamma, \varepsilon_{\text{opt}}(A, B) + \gamma]$, they correctly answer a query, and for $\varepsilon \in [\varepsilon_{\text{opt}}(A, B) - \gamma, \varepsilon_{\text{opt}}(A, B) + \gamma]$, they either answer correctly or choose not to answer. Our presentation culminated with an algorithm for the case where an arbitrary isometry may be performed on one of the sets, but we also obtained more efficient algorithms for the special cases of the isometry restricted to a translation only or a rotation only. In each case we also gave an algorithm with running time dependent on δ/γ and linear in n in the worst-case, implying that these (γ, γ)-approximate algorithms are especially efficient on dense data.

A primary strength of a (γ, γ)-approximate algorithm is that an incorrect answer is never returned. Algorithms that work with perturbed data must have some imprecision, but our algorithms have the attractive feature of being prudent enough not to answer when the query is too difficult. The imprecision inherent in a (γ, γ)-approximate algorithm is exhibited in the indecision interval, $[\varepsilon_{\text{opt}}(A, B) - \gamma, \varepsilon_{\text{opt}}(A, B) + \gamma]$.

Another attraction of the (γ, γ)-approximate algorithms presented in this paper is that the user is offered a trade-off between precision and run-time. We mention a manner whereby this trade-off can be used to remove the indecision of an approximate decision algorithm. If we wish to test for ε-congruence (for any isometry class) for a given value ε, we likely will first choose to execute our algorithm with a relatively large value of γ, since for each of our algorithms, the run-time is inversely proportional to an exponent of γ. If ε lies in the indecision interval $[\varepsilon_{\text{opt}}(A, B) - \gamma, \varepsilon_{\text{opt}}(A, B) + \gamma]$, then we may discover that our choice of γ was too large, for we

may not be given an answer. If we alternate setting $\gamma \leftarrow \gamma/2$ and repeating the test until receiving an answer, the cost of our procedure will be dominated by the final test. The effect of our approach is that the total run-time is within a constant factor of a single test which chooses for γ the "optimal" choice of $|\varepsilon_{\text{opt}}(A, B) - \varepsilon|$. This means that we can replace γ by $K_\varepsilon = |\varepsilon_{\text{opt}}(A, B) - \varepsilon|$ in the time bounds of our algorithms, and in the process obtain complete, not approximate, algorithms:

Theorem 5. *There exist decision algorithms for testing ε-congruence under translation with time-complexity $O(n^{1.5}(\varepsilon/K_\varepsilon)^4)$, under rotation with time-complexity $O(n^{2.5}(\varepsilon/K_\varepsilon)^3)$, and under general isometry with time-complexity $O(n^{2.5}(\varepsilon/K_\varepsilon)^5)$ (we can give alternate bounds for rotation and general isometry in the case of dense data).* Here, $K_\varepsilon = |\varepsilon_{\text{opt}}(A, B) - \varepsilon|$.

These new expressions agree with our intuition, since we believe that testing for ε-congruence should be harder when ε is close to $\varepsilon_{\text{opt}}(A, B)$. We can also adapt our methods to estimate $\varepsilon_{\text{opt}}(A, B)$ through a search procedure, where the tolerance of the estimate replaces γ in the time bound of the algorithm.

Known decision algorithms for ε-congruence are expensive, and therefore unsuitable for many applications. Our response to this situation has been to develop approximate decision algorithms, which enjoy substantially improved time bounds by perturbing the point sets in order to impose degenerate structure. Our algorithms never return an incorrect answer, and have bounds on the query values for which they can choose to give no answer. We feel that this combination of speed, correctness, and bounded imprecision characterizes our methods as practical algorithms for point matching.

References

[1] R.K. Ahuja, T.L. Magnanti and J.B. Orlin, Network flows, in: G.L. Nemhauser, A.H.G. Rinnooy Kan and M.J. Todd, eds., Handbook in Operations Research and Management, Vol. 1: Optimization (North-Holland, Amsterdam, 1990) 211–360.

[2] H. Alt, K. Mehlhorn, H. Wagener and E. Welzl, Congruence, similarity, and symmetries of geometric objects, Discrete Comput. Geom. (1988) 237–256.

[3] E.M. Arkin, K. Kedem, J.S.B. Mitchell, J. Sprinzak and M. Werman, Matching points into pairwise-disjoint noise regions: combinatorial bounds and algorithms. ORSA J. Comput. 4, (1992) 375–386.

[4] E.M. Arkin, J.S.B. Mitchell and K. Zikan, Algorithms for generalized matching of point sets, Manuscript, presented at the CORS/ORSA/TIMS National Meeting, Vancouver, B.C., May 1989.

[5] H.S. Baird, Model-Based Image Matching Using Location (MIT Press, Cambridge, MA, 1984).

[6] B. Behrends, Algorithmen zur Erkennung der ε-Kongruenz von Punktmengen, Diplomarbeit, Freie Universität Berlin, Germany, 1990.

[7] L.P. Chew and K. Kedem, Improvements on geometric pattern matching problems, in: SWAT'92, LNCS, Vol 621 (Springer, Berlin, 1992).

[8] T. Feder and R. Motwani, Clique partitions, graph compression and speeding-up algorithms, in: Proceedings of the 23rd Symposium on Theory of Computing (1991) 123–133.

[9] A.V. Goldberg, E. Tardos and R.E. Tarjan, Network flow algorithms, in: B. Korte, L. Lovász, H.J. Prömel and A. Schrijver, eds., Paths, Flows, and VLSI-Layout (Springer, Berlin, 1990) 101–164.

[10] A.V. Goldberg and R.E. Tarjan, A new approach to the maximum flow problem, J. the ACM 35 (1988) 921–940.
[11] P.J. Heffernan, The translation square map and approximate congruence, Inform. Process. Lett. 39 (1991) 153–159.
[12] D.P. Huttenlocher and K. Kedem, Efficiently computing the Hausdorff distance for point sets under translation, in: Proc. of the 6th ACM Symp. on Computational Geometry (1990) 340–349.
[13] D.P. Huttenlocher, K. Kedem and J.M. Kleinberg, On dynamic Voronoi diagrams and the minimum Hausdorff distance for point sets under Euclidian motion in the plane, in: proc. of the 8th ACM Symp. on Computational Geometry (1992) 110–119.
[14] D.P. Huttenlocher, K. Kedem and M. Sharir, The upper envelope of Voronoi-surfaces and its applications, in: Proc. of the 7th ACM Comp. Geom. Conference (1991) 194–203.
[15] K. Imai, S. Sumino and H. Imai, Minimax geometric fitting of two corresponding sets of points, in: Proc. of the 5th ACM Symp. on Computational Geometry, Saarbrücken, Germany (1989) 276–282.
[16] S. Iwanowski, Approximate Congruence and Symmetry Detection in the Plane, Ph.D. Thesis, Fachbereich Mathematik, Freie Universität Berlin, 1990.
[17] H.P. Lenhof and M. Smid, An optimal construction method for generalized convex layers, in: ISA'91 Algorithms Vol. 557 (Springer, Berlin, 1991) 349–363.
[18] G.E. Martin, Transformation Geometry (Springer, Berlin, LNCS, 1982).
[19] K. Mehlhorn, Data Structures and Algorithms 2: Graph Algorithms and NP-completeness (Springer, Berlin, 1984).
[20] F. Preparata and M.I. Shamos, Computational Geometry (Springer, Berlin, 1985).
[21] S. Schirra, Über die Bitkomplexität der ε-Kongruenz, Diplomarbeit, Fachbereich Informatik, Universität des Saarlandes, Saarbrücken, Germany, 1988.
[22] S. Schirra, Approximate decision algorithms for approximate congruence, Inform. Process. Lett. 43 (1992) 29–34.
[23] D. Sleator, An O (n log n) algorithm for maximum network flow, Technical Report STAN-CS-80-831, Computer Science Dept., Stanford University, 1980.
[24] D.D. Sleator and R.E Tarjan, A data structure for dynamic trees, in: Proceedings of the 13th Symposium on Theory of Computing (1981) 114–122.
[25] J. Sprinzak, Master's Thesis, Dept. of Computer Science, Hebrew University Jerusalem, 1990.
[26] R.E. Tarjan, Data Structures and Network Algorithms, Vol. 44 of CBMS-NSF Regional Conference Series in Applied Mathematics (SIAM, Philadelphia, PA, 1983).